

Estendere OpenOffice.org con Java

*Percorso di uno sviluppatore Java per
estendere OpenOffice.org.*

Giuseppe Castagno
Sviluppatore volontario di
OpenOffice.org
beppec56@openoffice.org

14:00 - 19:00
5/11/2009

Benvenuti !



OpenOffice.org
CONFERENCE 2009 ORVIETO

Agenda



- Parte 1: Un minimo di teoria...
 - Introduzione
 - Documentazione
 - Strumenti di sviluppo (IDE)
 - Java con OpenOffice.org
- Intervallo



Agenda



- Parte 2: e ora scriviamo il codice!
 - Scrittura di un nuovo componente UNO
 - Debug Java in OpenOffice.org
 - Esempi di Java in OpenOffice.org
 - Chiedere aiuto
- Infine, le domande finali



Agenda



- Parte 1: Un minimo di teoria...
 - **Introduzione**
 - Documentazione
 - Strumenti di sviluppo (IDE)
 - Java con OpenOffice.org



Introduzione: automatizzare



- Automatizzare un'applicazione
 - Molti significati, dipende dal contesto:
per aggiornamento automatico per esempio
Oppure:
 - Per facilitare il compito all'utente
 - Per azioni ripetitive
- Si può realizzare con macro in StarBasic
 - Registrare 'al volo' o scritte per l'occasione



Introduzione: estendere



- Estendere un'applicazione
 - Rendere persistente una sequenza automatica
 - Oppure:
 - Aggiungere funzionalità non presenti
 - Per es. firma digitale custom
 - Questo sarà il nostro esempio
- Si realizza con le estensioni
 - StarBasic, Java, Python, C++



Intro: estendere con Java



- Uso di Java nelle estensioni
 - Molto simile al C++ (scelta mia personale !)
 - Developer's Guide ricca di esempi
 - Può servire come intro al codice 'core' di OOo
 - Disponibili 'helper' per registrazione classi e
 - classi base per componenti UNO
- Focus su Java nelle estensioni



Intro: estendere con Java



- Il progetto Java di supporto alla presentazione
 - Home progetto a:
<http://ooo-xadessig-it.forge.osor.eu/it>
 - Svn 'checkout' anonimo:
<https://svn.forge.osor.eu/svn/ooo-xadessig-it/branches/oooconf2009-demo>
- Potete scaricare un 'workspace' Eclipse da qui:
<http://www.acca-esse.it/dwnld/ooohs/oxsit-conf-branch-download.zip>



Agenda



- Parte 1: Un minimo di teoria...
 - Introduzione
 - **Documentazione**
 - Strumenti di sviluppo (IDE)
 - Java con OpenOffice.org



Documentazione: siti web



- Wiki su estensioni: Extensions
 - Da leggere, anche in italiano
 - Introduce le estensioni
 - Ha link ad articoli molto utili
- La Developer's Guide
- Il progetto api
 - In cui trovate la documentazione API



Documentazione: DevGuide



- La guida allo sviluppatore
 - Il documento principe !
 - In formato wiki:
<http://wiki.services.openoffice.org/wiki/Documentation/De>
- Potete stampare PDF o scaricare ODF.



Documentazione: API

- La documentazione API
 - Descrive servizi e interfacce già disponibili
 - In formato html:
<http://api.openoffice.org/docs/common/ref/com/sun/star/>
- La consulterete spesso !



Documentazione: SDK



- La SDK di OOo
 - Necessaria per lo sviluppo in Java
 - Contiene classi Java utili per lo sviluppatore
 - Obbligatoria per nuove interfacce e componenti UNO



Documentazione: UNO

- UNO: ma cos'è?
 - UNO significa Universal Network Object
 - Fornisce un ambiente multiplatforma,
 - intra-rete, e
 - connette linguaggi di programmazione diversi
- Definizione ufficiale qui:
<http://wiki.services.openoffice.org/wiki/Documentatio>



Documentazione: UNO



- UNO: riassunto della guida
 - Linguaggio di definizione simile a CORBA IDL
 - Componenti creati dal service manager di OOo secondo quanto presente nella 'OOo registry'
 - Interfacce definite in UNOIDL per comunicazione tra componenti



Documentazione: UNO

- UNO: riassunto della guida
 - La comunicazione è trasparente allo sviluppatore
 - Gli oggetti di OOo comunicano in un ambiente UNO (OOo API è la specifica), con rete o altro mezzo
 - Serve la libreria specifica del linguaggio (bridge)



Documentazione: UNO



- UNO: nuovi componenti
 - Possibili solo in Java, Python o C++
 - In StarBasic non si possono creare nuovi componenti
 - Ma potete usare quelli nuovi da voi creati



Documentazione: UNO

- UNO: nuovi componenti come?
 - Specificare
 - Usato il linguaggio UNO-IDL per la descrizione delle nuove interfacce (opzionale)
 - Implementare
 - Il codice (Java nel nostro caso) che realizza la specifica
 - Generare
 - Con la OOo-SDK



Documentazione: estensione



- Ma una estensione, come è fatta?
 - Essenzialmente un package zip, di tipo 'oxt'
 - Formazione predefinita, file obbligatori:
 - META-INF/manifest.xml
 - description.xml
- Esempio: `oxsit-ext_conf`



Estensione: aggiornare

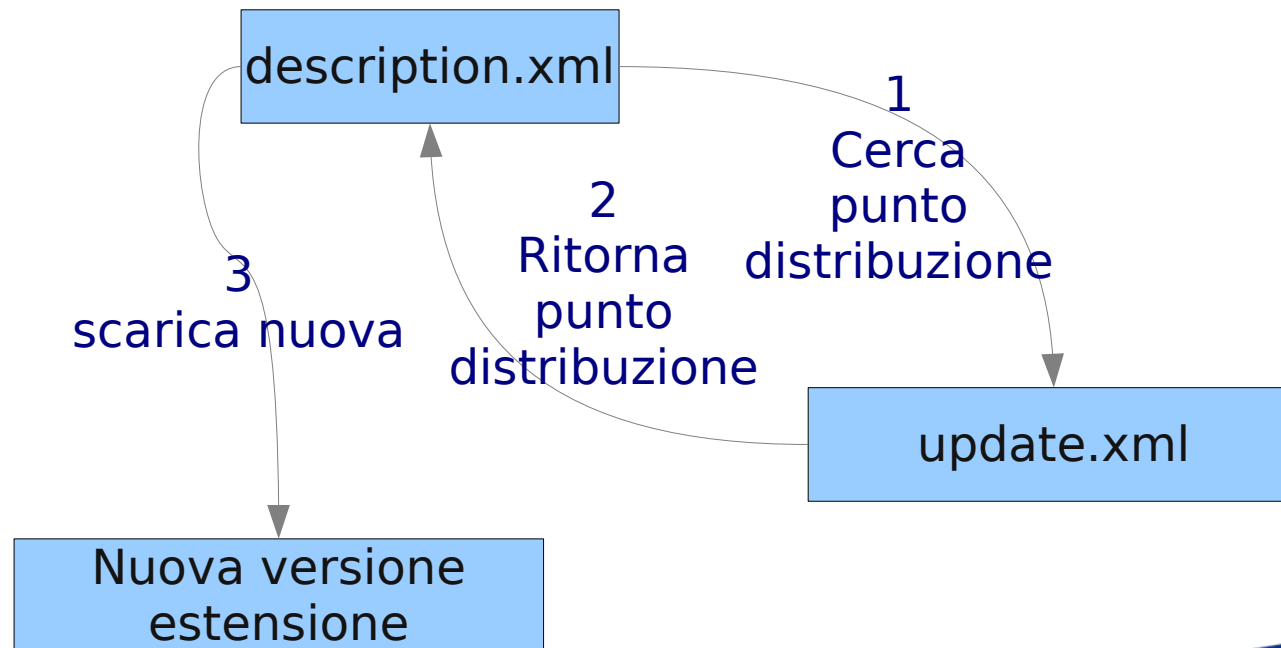


- Aggiornamento in linea di una estensione
 - Può essere reso semiautomatico
 - In description.xml informazioni sul target
 - Il target, un file xml per informazioni su punto di distribuzione
 - Infine, scarica nuova da distribuzione



Estensione: aggiornare

- Sicurezza upgrading
 - 'Tampering' possibile



Estensione: aggiornare



- Esempio in: `oxsit-ext_conf`
 - Anche se disattivato
- Dettagli: DevGuide
Extensions



AddOn, AddIn

- Significato:
- Per entrambi, aggiunta di funzionalità, poi:
- AddOn
 - Estensione che modifica la GUI
- AddIn
 - Estensione che non modifica la GUI



Agenda



- Parte 1: Un minimo di teoria...
 - Introduzione
 - Documentazione
 - **Strumenti di sviluppo (IDE)**
 - Java con OpenOffice.org



Strumenti di sviluppo (IDE)



- Netbeans
 - Possibilità di plugin per sviluppo rapido
info sul wiki di OOo:
http://wiki.services.openoffice.org/wiki/OpenOffice_NetBeans_I
- Eclipse
 - Possibilità di plugin per sviluppo rapido
info sul wiki di OOo:
<http://wiki.services.openoffice.org/wiki/JavaEclipseTuto>
 - Utile per file IDL



Strumenti di sviluppo (IDE)



- Uso di Eclipse che si descriverà
 - Non usa plugin
 - Usa ant per la compilazione e il 'build'
 - Editor interni di Eclipse per codice e file di configurazione
 - Se plugin per OOO installato, anche editor per IDL (evidenzia sintassi)



Strumenti di sviluppo (IDE)



- Pro
 - Indipendente
 - Controllo fine sulle possibilità di aggiornamento remoto
- Contro
 - Può sembrare macchinoso
 - Necessaria conoscenza struttura estensioni



Agenda



- Parte 1: Un minimo di teoria...
 - Introduzione
 - Documentazione
 - Strumenti di sviluppo (IDE)
 - **Java con OpenOffice.org**



Java con OpenOffice.org



- Due modi di utilizzo
 - Come applicazione: OOo è un servizio
 - Nelle estensioni, diventando parte di OOo.
- Il nostro focus sarà su Java nelle estensioni
 - La funzionalità di connessione è dettagliata nella Developer's Guide
 - L'uso di Java nelle estensioni un po' meno...



Java con OpenOffice.org



- Come applicazione che usa OOo come servizio
 - La connessione è tramite TCP
 - Un 'port' TCP è dedicato allo scopo
 - Necessario avviare OOo in modo specifico
 - O 'aprire' permanentemente il 'port'.



Java, OpenOffice.org servizio



- OOo si avvia con in più sulla linea di comando
 - accept=' socket , host=0 , port=9000 ; urp ; '
- In Java apposite classi 'helper' permettono la connessione
 - Servono anche alcune librerie (jar) nella distribuzione OOo (bootstrap)



Java, OpenOffice.org servizio



- Informazioni in Developer's Guide:

<http://wiki.services.openoffice.org/wiki/Documentation/DevGuide/>

- Il seguito della DevGuide spiega in dettaglio come procedere



Java con OpenOffice.org



- Qualche parola sulla GUI
 - È consigliabile la GUI di OOo
 - Swing può essere problematica
- Problemi dovuti ai thread
- Un UNO singleton forse?
 - Soluzione da provare !



Java con OpenOffice.org



- Il progetto Java di supporto alla presentazione
 - Funzionalità firma digitale XAdES secondo legge italiana
 - Contiene le soluzioni ai problemi più comuni
 - E anche non comuni (UNO singleton, UNO logger custom, configurazione)
 - Nuove interfacce UNO e nuovi componenti



Java con OpenOffice.org



- Il progetto Java di supporto alla presentazione
 - Riferimenti frequenti nel seguito sessione
 - Demo veloce dell'estensione....



OpenOffice.org SDK



- Installazione della SDK
 - È necessaria
 - Scaricatela dal mirror più vicino a voi.
<http://download.openoffice.org/sdk>
- Seguite le istruzioni di installazione fornite



OpenOffice.org SDK



- A cosa serve?
 - Fornisce classi utili (in GNU/Linux: /opt/openoffice.org/basis3.1/sdk/classes)
 - Molti esempi utili (in GNU/Linux: /opt/openoffice.org/basis3.1/sdk/examples)
 - Obbligatoria per nuove interfacce UNO



OpenOffice.org SDK

- Cosa fa
 - Ricava la specifica compilata:
 - `idlc` → `regmerge` → `oxsit-uno_types.uno.rdb`.
 - Dalla specifica ricava le classi Java binarie
 - `oxsit-uno_types.uno.rdb` → `javamaker` → `.class`
 - Dettagliato nella seconda parte
 - progetto `oxsit-uno_types`



Java con OpenOffice.org



- Riassumendo, per Java servono:
 - OpenOffice.org installato
 - La documentazione API (in Internet)
 - La SDK di OpenOffice.org
 - Una Java IDE
 - Suggesto plugin OOo per Eclipse (facilita editing IDL)



Java con OpenOffice.org



- Librerie: per Java servono:
 - Da OpenOffice.org installato le librerie:
/opt/openoffice.org/ure/share/java/juh.jar:
/opt/openoffice.org/ure/share/java/jurt.jar
/opt/openoffice.org/ure/share/java/ridl.jar
/opt/openoffice.org/basis3.1/program/classes/unoil.jar
 - I path di GNU/Linux, OOo 3.1.1



Agenda

- Intervallo
 - E sì, ci vuole...



Agenda



- Parte 2: e ora scriviamo il codice!
 - **Scrittura di un nuovo componente UNO**
 - Debug di Java in OpenOffice.org
 - Esempi di Java in OpenOffice.org
 - Chiedere aiuto



Nuovo componente UNO



- UNO: possibilità:
 - C'è già una interfaccia
 - Non esiste interfaccia adatta
 - Una procedura specifica lo richiede
 - Le interfacce esistenti sono 'unpublished'
- DevGuide: l'intera sezione Writing UNO Components



Nuovo UNO: c'è interfaccia

- Classe Java → componente UNO (servizio)
- Interfacce base con l'aiuto di 'helper class'
- Interfaccia specifica da implementare
 - Una delle due classi disponibili:
`com.sun.star.lib.uno.helper.ComponentBase`
`com.sun.star.lib.uno.helper.WeakBase`
 - Implementate le interfacce
- DevGuide:
Class Definition with Helper Class



Nuovo UNO: c'è interfaccia

- Esempio: `it.plio.ext.oxsit.comp.SyncJob`
- Sezione nella DevGuide:
Core Interfaces to Implement



Nuovo UNO: registrazione



- Tramite metodo di supporto:

```
public synchronized static boolean  
__writeRegistryServiceInfo( XRegistryKey xRegistryKey )
```

- Path classe in manifest (Java):

```
<attribute name="RegistrationClassName"  
value="it.plio.ext.oxsit.RegisterServices" />
```

- Sezione nella DevGuide:

Write Registration Info Using Helper Method



Nuovo UNO: creazione

- Factory resa pubblica con 'helper':

```
public synchronized static
XSingleComponentFactory
__GetComponentFactory( String sImplementationName
)
```

- Sezione nella DevGuide:

Providing a Single Factory Using Helper Method



Nuovo UNO: installazione



- Al momento dell'installazione (unopkg)
 - Registrazione del componente
 - Registrazione della factory



Nuovo UNO: non c'è interfaccia



- Necessario implementare interfaccia
 - 1) descrivere interfaccia (UNO-IDL)
 - 2) con OooSDK ottenere le classi
 - 3) preparare jar e rdb da usare in installazione
- Jar usato per sviluppo nuovo componente



Nuovo UNO: nuova interfaccia



- Oltre ai passi precedenti in più la nuova interfaccia
- Descrizione UNO-IDL con text editor
 - Specifiche UNO-IDL in DevGuide:
Appendix D: UNOIDL Syntax Specification
- Esempi in `oxsit-uno_types/idl`



Nuovo UNO: nuova interfaccia

- Compilare in SDK:
 - Ricava la specifica compilata:
 - `idlc` → `regmerge` → `oxsit-uno_types.uno.rdb`.
 - Dalla specifica ricava le classi Java binarie
 - `oxsit-uno_types.uno.rdb` → `javamaker` → `.class`
- Poi i file `*.class` si inseriscono in jar
- No valori speciali in manifest



Nuovo UNO: nuova interfaccia



- Esempio in `oxsit-uno_types`
 - Script ant indicato `oxsit-uno_types/build.xml`
 - Script bash `oxsit-uno_types/compile-idl.sh`
 - Adattate il file:
`oxsit-uno_types/build.Linux-i386.properties.xml`
 - Il valore di `oo_sdk_init_script` dipende da ambiente
 - Poi:

```
ant prebuild
ant build-jar
```



Nuovo UNO: nuova interfaccia

- E poi? Come installo ?
- IN META-INF/manifest.xml:

```
<manifest:file-entry manifest:full-path="oxsit-uno_types.uno.rdb" manifest:media-type="application/vnd.sun.star.uno-typelibrary;type=RDB"/>
```

```
<manifest:file-entry manifest:full-path="oxsit-uno_types.uno.jar" manifest:media-type="application/vnd.sun.star.uno-typelibrary;type=Java"/>
```



Nuovo UNO: nuova interfaccia



- Esempio:

`oxsit-ext_conf/extension/META-INF/manifest.xml`



Agenda



- Parte 2: e ora scriviamo il codice!
 - Scrittura di un nuovo componente UNO
 - **Debug di Java in OpenOffice.org**
 - Esempi di Java in OpenOffice.org
 - Chiedere aiuto



Debug Java in OpenOffice.org



- Debug di applicazione Java esterna, OOo come servizio
 - Debug come applicazione Java, differente avvio di OOo, aprire la 'porta' TCP per la connessione: aggiungere parametro
`-accept='socket,host=0,port=9000;urp;`
nella linea di comando
- Debug di estensione OOo in Java
 - È un po' più macchinoso, spiegato in dettaglio nelle pagine successive



Debug di estensione OOo in Java



- Installate OOo a parte
- Abilitate per il debug remoto
- Installate estensione in corso di debug



Debug di estensione OOo in Java



- Installate OOo a parte
 - Non rovinare installazione standard
 - L'estensione in sviluppo può non funzionare correttamente distruggendo la vostra installazione
 - Si fa riferimento al progetto Java di supporto per i dettagli



Installate OOo a parte

- Per versione GNU/Linux, potete usare come esempio il file bash:

```
oxsit-scratch/bash-scripts/install-ooo-binary.sh
```

- Se il vostro progetto lo richiede, anche un locale in più, esempio il file bash:

```
oxsit-scratch/bash-scripts/install-ooo-lang-binary.sh
```



Installate OOo a parte

- Preparate OOo per una configurazione utente diversa
- Due metodi possibili:
 - Aggiungete in linea di comando (riga unica):
-env:UserInstallation=
\\\$SYSUSERCONFIG/.ooo-test-ext
 - Modificate il file
bootstraprc in:
<ooo-inst-root>/openoffice.org3/program



Installate OOo a parte

- Modificate il file bootstraprc in:

`<ooo-inst-root>/openoffice.org3/program`

- Cambiate la linea:

`UserInstallation=$SYSUSERCONFIG/.openoffice.org/3`
per esempio, in:

`UserInstallation=$SYSUSERCONFIG/.ooo-test-ext`



Abilitate per il debug remoto

- Si abilita Java, con 'porta' TCP dinamica
 - La configurazione fissa disturba l'installazione
- Avviate OOo di debug con (GNU/Linux):

```
cd <ooo-inst-root>/openoffice.org3/program  
./soffice
```
- OOo richiede info utente come al primo avvio.



Abilitate per il debug remoto



- Con il comando da menu:

Strumenti > Opzioni > Java

aggiungete la seguente linea, con il pulsante 'Parametri' (una sola linea!):

```
-agentlib:jdwp=transport=dt_socket,server=y,suspend=n
```

- Pagina seguente per figure



Abilitate per il debug remoto

The screenshot shows the OpenOffice.org Writer interface with the 'Opzioni - OpenOffice.org - Java' dialog box open. The 'Opzioni Java' section is selected, and the 'Usa un ambiente runtime Java' checkbox is checked. Below this, a table lists installed Java Runtime Environments (JREs):

Fornitore	Versione	Caratteristiche
Sun Microsystems Inc.	1.6.0_16	

The 'Parametri di avvio Java' dialog box is also open, showing the 'Parametro di avvio Java' field and the 'Parametri di avvio assegnati' field. The assigned parameters are: `-agentlib:jdwp=transport=dt_socket,server=y,suspend=n`. Below the field, an example is provided: `Ad esempio: -Dmyprop=c:\programmi\java`. The dialog has 'OK', 'Annulla', and '?' buttons.

Abilitate per il debug remoto



- Java assegna ogni volta una porta diversa
- Per vedere quale (GNU/Linux):

```
netstat -tulpn | grep soffice.bin
```
- Di solito è un numero alto
- In Eclipse creazione di una configurazione debug remoto
 - Necessario cambiare ogni volta numero porta.



Agenda

- Parte 2: e ora scriviamo il codice!
 - Scrittura di un nuovo componente UNO
 - Debug di Java in OpenOffice.org
 - **Esempi di Java in OpenOffice.org**
 - Chiedere aiuto



Esempi di Java in OOo

- La OooSDK:
`/opt/openoffice.org3/basis-link/sdk/examples`
- Il codice OOo? Main browser a:
`http://svn.services.openoffice.org/opengrok`
- File di configurazione estensioni
 - Descritti a:
 - `officecfg/registry/schema/org/openoffice/Office/Addons.xcs`



Esempi di Java in OOO

- File dei menu
 - Vari e in diverse posizioni, menu generale a:
`officecfg/registry/data/org/openoffice/Office/UI/GenericCommands`
 - Di Writer a:
`officecfg/registry/data/org/openoffice/Office/UI/WriterCommands.`
- Alcuni wizard (proced. guidate):
`wizards/com/sun/star/wizards`



Esempi di Java in OOO

- Libreria Java usata
 - Jurt.jar:
jurt/com/sun/star
- Classi base per componenti
javaunohelper/com/sun/star/lib/uno/helper
-



Agenda



- Parte 2: e ora scriviamo il codice!
 - Scrittura di un nuovo componente UNO
 - Debug di Java in OpenOffice.org
 - Esempi di Java in OpenOffice.org
 - **Chiedere aiuto**



Mailing list sviluppatori



- Mailing list:
 - dev@openoffice.org principale sviluppatori, molto trafficata
 - dev@api.openoffice.org domande specifiche API, molto trafficata
 - dev@extensions.openoffice.org Specifica per le estensioni
 - dev@it.openoffice.org lista sviluppatori italiana
- Infine l'autore:
 - beppe56@openoffice.org



Siti web



- La Developer's Guide
- Il progetto api
 - In cui trovate la documentazione API
- Wiki su estensioni: Extensions



Estendere OpenOffice.org (Java)

Percorso di uno sviluppatore Java per estendere OpenOffice.org.

Giuseppe Castagno
Sviluppatore volontario
di OpenOffice.org

14:00 - 19:00
5/11/2009

Grazie dell'attenzione ! E ora le domande...



Licenza

Questa presentazione è concessa in licenza in accordo a:

Attribuzione-Non commerciale-Condividi
allo stesso modo 2.5 Italia

Giuseppe Castagno

beppec56@openoffice.org

